```
"OTN" => "OTN__object_title_name",
  "OTT" => "OTT__title_type ",
  "OTY" => "OTY__object_type",
  "RDD" => "RDD__related_document_description",
  "RDG" => "RDG_related_documents",
  "RDL" => "RDL__related_document_identifier_link",
  "RDR" => "RDR__related_document_relationship_type",
  "RID" => "RID__related_image_description",
  "RIG" => "RIG_related_images",
  "RIL" => "RIL__related_image_identifier_link",
  "RIP" => "RIP__related_image_preferred",
  "RIR" => "RIR__related_image_relationship_type",
  "RMD" => "RMD__related_multimedia_description",
  "RMG" => "RMG_related_multimedia",
  "RML" => "RML__related_multimedia_identifier_link",
  "RMR" => "RMR__related_multimedia_relationship_type",
  "RWD" => "RWD__related_works_description",
  "RWG" => "RWG_related_works_of_art",
  "RWL" => "RWL__related_works_identifier_link",
  "RWR" => "RWR__related_works_relationship_type ",
  "STD" => "STD__style_period_description",
  "STG" => "STG_style_period",
  "STT" => "STT__style_period_terms ",
  "SUG" => "SUG_subject_matter",
  "SUI" => "SUI__subject_matter_iconography",
  "SUP" => "SUP__subject_matter_preiconographic_description",
  "SUT" => "SUT__subject_matter_index_terms ",
  "XAM" => "XAM__amico_mode",
  "XCC" => "XCC__dc_creator_corporatename",
  "XCM" => "XCM__amico_format_colormetric",
  "XCN" => "XCN_dc_creator",
  "XCP" => "XCP__dc_creator_personalname",
  "XCR" => "XCR__dc_creator_role ",
  "XDA" => "XDA__dc_date",
  "XDC" => "XDC__dc_contributor_corporatename",
  "XDE" => "XDE__dc_description",
  "XDL" => "XDL__metadata_delition_flag",
  "XDN" => "XDN_dc_contributor",
  "XDP" => "XDP__dc_contributor_personalname",
  "XDR" => "XDR__dc_contributor_role ",
  "XFC" => "XFC__amico_format_compression",
  "XFD" => "XFD__amico_format_dimensions",
  "XFE" => "XFE__amico_format_encoding",
  "XFF" => "XFF__amico_format_filesize",
  "XFO" => "XFO_dc_format",
  "XFP" => "XFP__amico_format_colorpalette",
  "XID" => "XID__dc_resource_identifier",
  "XLY" => "XLY__metadata_library_year",
  "XMN" => "XMN__amico_media_note",
  "XPR" => "XPR__metadata_data_processing_note",
  "XPU" => "XPU__dc_publisher",
```

FIGURE 35C

```
"XRI" => "XRI__dc_relation_identifier",
  "XRS" => "XRS__dc_rights",
  "XRT" => "XRT__dc_resourcetype",
  "XRY" => "XRY__dc_relation_type",
  "XTI" => "XTI__dc_title",
  "XVD" => "XVD__amico_metadata_validation_date",
  "XVV" => "XVV__amico_data_dictionary_version"
 );

sub name {           # long or short tagnames
 return $_[0]                # short
#  return $long_tag{$_[0]}  # long
}

my %is_group = ();  # what tags are groups?

while ((my $tag, my $long) = each %long_tag) {
 $is_group{$tag} = 1 if ($long !~ /__/);
}


my %group_members =    # members of each group
 (
  "CLG" => "CLS CLT",
  "CRG" => "CRQ CRT CRN CRC CDT CBD CBP CBQ CDD CDP CDQ CAD CAP CGN CRB CRR
CNO",
  "CXG" => "CXD CXP CXS CXT",
  "DCG" => "DCB DCD",
  "MEG" => "MCM MED MDV MDU MEQ",
  "OCG" => "OCT OCS OCE OCQ",
  "OMG" => "OMD OMT OMM OMS",
  "OOG" => "OON OOP OOA OOC",
  "ORG" => "ORS ORL",
  "OTG" => "OTN OTT",
  "RDG" => "RDD RDR RDL",
  "RIG" => "RIP RID RIR RIL",
  "RMG" => "RMD RMR RML",
  "RWG" => "RW,D RWR RWL",
  "STG" => "STD STT",
  "SUG" => "SUP SUI SUT",
  "XCN" => "XCP XCC XCR",
  "XDN" => "XDP XDC XDR",
  "XFO" => "XFE XFP XCM XFD XFF XFC",
  "XRE" => "XRY XRI"
 );
```

FIGURE 35D

```
my %group = ();             # inverse: returns the group of a member

while ((my $group, my $memstr) = each %group_members) {
  my @members = ($memstr =~ m/[A-Z]{3}/g);
  foreach (@members) {
    $group{$_} = $group;
  }
}


sub escXML {                # escape characters: EXTEND/FIX!
  $_ = $_[0];
  s/</&lt;/g;
  s/&/&amp;/g;
  s/á/a/g;
  s/à/a/g;
  s/â/a/g;
  s/é/e/g;
  s/è/e/g;
  s/ñ/n/g;
  $_;
}

sub output_fields {  # output all fields from current record
  @record = @_;
  my $field = shift @record;         # get the next field
  if (defined $field) {
    &output_field($field);           # output it (and more if group!)
    &output_fields(@record);                # recurse through the remains
                                # of @record
  }
}

sub output_field {   # output the given field PLUS follow-up group
                     # members!
  (my $field) = @_;
  if ($field =~ /([A-Z]{3})(.*)/) {      # is it a 3 letter tag + data?
    my $tag = $1; my $data = $2;
    if ($long_tag{$tag}) {               # do we know this tag?
      if ($is_group{$tag}) {             # is it a group?
         print &ind, "<", &name($tag), ">\n";
         $tab +=2 ;
         &output_group($tag);                  # and output closing tag!
      }
      else {                             # it's a simple recognized tag
         print  &ind, "<", &name($tag), ">",
             &escXML($data),
             "</", &name($tag), ">\n";
      }
    }    else {                          # don't know this tag!
     print  "<am_ERROR type = \"unrecognized tag\">";
     print  &ind, "<$tag>", &escXML($data), "</$tag>\n";
     print  "</am_ERROR>\n";
    }
```

FIGURE 35E

```
} else {                            # don't understand that field!
    print  "<am_ERROR type = \"unrecognized field\">", $_;
    print  "</am_ERROR>\n";
  }
}

sub output_group {  # output fields WHILE in same group
  (my $grp) = @_;
  my $field = shift @record;        # get the next field
  if (defined $field) {
    if ($field =~ /([A-Z]{3})/) {# should be a tag
      if (defined $group{$1} and $group{$1} eq $grp) {
                                    # still in the same group??
        &output_field($field);
        &output_group($grp);              # STAY in the same group
      } else {                      # LEAVE group!
        $tab -= 2;
        print  &ind, "</", &name($grp), ">\n";      # closing tag for group
        &output_field($field);
      }
    } else {                        # don't understand that field!
      print  "<am_ERROR type = \"unrecognized field\">", $_;
      print  "</am_ERROR>\n";
    }
  }                                 # empty @record => done
}


sub get_record {      # read the next record and return first tag
  chomp;
  @record = split /\}~/;            # End-Of-Record = "|\n"
  my $last = pop(@record);          # get EOR
  if ($last ne "|") {               # ...and check
    print STDOUT "*** ERROR: unknown delimiter <$last>\n";
  }
  if (@record) {        # check if non-empty
    return substr($record[0],0,3)    # return the tag name
  } else {
    print STDOUT "*** WARNING: empty record\n";
    return 0
  }
}
```

FIGURE 35F

```
#==========================================================================
#     M A I N
#==========================================================================

$| = 1;

open(AM_OBJS, ">amico_objects.xml") or die "*** ERROR <@_>\n<$?>\n";
open(AM_MEDIA, ">amico_media.xml") or die "*** ERROR <@_>\n<$?>\n";

print AM_OBJS <<EOF;
<am_objects>
EOF

print AM_MEDIA <<EOF;
<am_media>
EOF

# print AM_OBJS <<EOF;
# <?xml:stylesheet type="text/xsl" href="amico_objects.xsl"?>
# <am_objects>
# EOF

# print AM_MEDIA <<EOF;
# <?xml:stylesheet type="text/xsl" href="amico_media.xsl"?>
# <am_media>
# EOF

while (<>) {
  if (my $tag = &get_record) {              # get next record and tag name
    if ($tag eq "AID") {
      select(AM_OBJS);
      print &ind, "<am_object>\n";
      $tab += 2;
    &output_fields(@record) ;
      $tab -= 2;
      print &ind, "</am_object>\n";
    } elsif  ($tag eq "XID") {
      select(AM_MEDIA);
      print &ind, "<am_media_metadata>\n";
      $tab += 2;
      &output_fields(@record) ;
      $tab -= 2;
      print &ind, "</am_media_metadata>\n";
    } else {
      print STDOUT "*** ERROR: unknown tag <$tag> in record: <@record>\n";
    }
  } else {
    print STDOUT "*** ERROR: get_record failed\n";
  }
}


print AM_OBJS "</am_objects>\n";
print AM_MEDIA "</am_media>\n";
```

FIGURE 35G

```
#---------------------------------------
# Perl Script to convert XML SLA version
# into software independent OAV
# representation, ready to be # loaded
# into a variety of engines:
#      - Prolog engine, or
#      - relational database engine, or
#      - XML database engine, or
#      - other
#
# SCRIPT devised by Richard Marciano &
#              Bertram Ludaescher &
#                     Reagan Moore
# August 20, 2000, copyright RiM + BL + ReM
#---------------------------------------

#!/usr/local/bin/perl
#use strict;

#-------------------------------------------------------------
@boa; %boatype_hash = (); my $bill_count = 0;
    my $amdt_count = 0; my $con_res_count = 0;
    my $j_res_count = 0; my $res_count = 0;

@abstract; %a_value = ();
@congressional_record; %cr_value = ();
@cosponsors; %cs_value = (); @date_introduced;
    %di_value = (); @digest; %d_value = ();
@latest_status; %ls_value = (); @status_actions;
    %sa_value = (); @official_title; %ot_value = ();
@sponsor; %s_value = (); @statement_of_purpose;
    %sop_value = (); @submitted_by; %sb_value = ();
@submitted_for; %sf_value = (); @filename_senator;
@filename_period; @prepared_by; @senator;
@occurrence_section; @occurrence_committee;

@topic_index; %ti_value= ();
#-------------------------------------------------------------

my $MORE_SIZE = 20;

my $bill_index_2 = "";
my %tempH = ();

my $line; my $h1; my $h2 = ""; my $h3; my $h4; my $h5; my $h6_1 = ""; my $h6_2
= "";
my $prev_h2 = "";
my $section = 0; my $committee = "";
my $senNAME = ""; my $state = ""; my $senid;

my $filename;
my $line_number;
```

36A	36N
36B	36O
36C	36P
36D	36Q
36E	36R
36F	36S
36G	36T
36H	36U
36I	36V
36J	36W
36K	36X
36L	36Y
36M	36Z

FIGURE 36A

```
my @allfile;

opendir THISDIR , "XMLDATA" or die "can't find DIRECTORY: $!";
@allfile = grep /_LAR/, readdir THISDIR;
closedir THISDIR;
open( LOG, ">logfile" ) || die "*ERROR: can't open logfile\n" ;

#==============================================================================
=============

foreach my $sen (@allfile) {


#my $sen = "D_$ARGV[1]_LAR$ARGV[0]_106.xml";
#my $sen = "D_1CP_LAR_S106_106.xml";
#my $sen = "D_1CP_LAR_S216_106.xml";
#my $sen = "D_1_LARI_S272_106.xml";

    $sen =~ m/.+\_.+\_.+\_S(\d+)\_.+/;
    $senid = $1;
    $senid = $1;
    $senNAME = "";

    $filename = $sen;
    $line_number = 0;

    open( SEN106, "XMLDATA/$sen" ) || die "*ERROR: can't open $sen\n" ;
    #open( SEN106, "$sen" ) || die "*ERROR: can't open $sen\n" ;

    &process_header( $sen );

 N1:while( $line = <SEN106> ) {
        $line_number++;
 N2:;
    # DETECT SECTION headers
    if ( $line =~ m/hidden="on">(.+)<\/string>/ ) {
        $h1 = $1;
# <p align="left" bold="on"><string bold="on">SECTION I. SPONSORED
MEASURES</string></p>
        if ( $h1 =~ m/SECTION I\./ ) {
            $section = 1; $h6_1 = ""; $h6_2 = "";

            my $ov = $senid . "_" . "$line_number";

            my $lên = $#occurrence_section;
            $occurrence_section[ $len + 1 ] = [ ($senid, $line_number,
$section, $h1) ];
        }
        elsif ( $h1 =~ m/SECTION II\./ ) {
            $section = 2; $h6_1 = ""; $h6_2 = "";

            my $ov = $senid . "_" . "$line_number";
```

FIGURE 36B

```
        elsif ( $h1 =~ m/SECTION IV\./ ) {
            $section = 4;

            my $ov = $senid . "_" . "$line_number";

            my $len = $#occurrence_section;
            $occurrence_section[ $len + 1 ] = [ ($senid, $line_number,
$section, $h1) ];
        }
        elsif ( $h1 =~ m/SECTION V\./ ) {
            $section = 5; $h6_1 = ""; $h6_2 = "";

            my $ov = $senid . "_" . "$line_number";

            my $len = $#occurrence_section;
            $occurrence_section[ $len + 1 ] = [ ($senid, $line_number,
$section, $h1) ];
        }
        elsif ( $h1 =~ m/SECTION VI\./ ) {
            $section = 6; $h6_1 = ""; $h6_2 = "";

            my $ov = $senid . "_" . "$line_number";

            my $len = $#occurrence_section;
            $occurrence_section[ $len + 1 ] = [ ($senid, $line_number,
$section, $h1) ];
        }
        elsif ( $h1 =~ m/SECTION VII\./ ) {
            $section = 7; $h6_1 = ""; $h6_2 = "";

            my $ov = $senid . "_" . "$line_number";

            my $len = $#occurrence_section;
            $occurrence_section[ $len + 1 ] = [ ($senid, $line_number,
$section, $h1) ];

            &process_index;
        }
        else {
            print LOG "!!! error:COULD NOT RECOGNIZE SECTION NUMBER !!!!\n";
        }
    }

    # DETECT **** BILL NUMBERS
    elsif ( $line =~ m/>\*\*\*\* (.+)<\/p>/ ) {
        $h2 = $1;
        $h2 =~ s/\s*//g;



        if ( $prev_h2 ne "" ) {
#           S.123           1 dot
#           S.Amdt.123   2 dots
```

FIGURE 36C

```
                $boatype_hash{ bill }{ $bill }++;
            }
        }
        else {
            if( $list[1] eq "Amdt" ) {
                $amdt_count++;

                my $bill;
                foreach $bill ( sort keys %{$tempH{$prev_h2}} ) {
                    if ( $boatype_hash{ amdt }{ $bill } eq "" ) {
                        $boatype_hash{ amdt }{ $bill } = 0;
                    }
                    $boatype_hash{ amdt }{ $bill }++;
                }
            }
            elsif( $list[1] eq "Con" ) {
                $con_res_count++;

                foreach my $bill ( sort keys %{$tempH{$prev_h2}} ) {
                    if ( $boatype_hash{ con_res }{ $bill } eq "" ) {
                        $boatype_hash{ con_res }{ $bill } = 0;
                    }
                    $boatype_hash{ con_res }{ $bill }++;
                }
            }
            elsif( $list[1] eq "J" ) {
                $j_res_count++;

                foreach my $bill ( sort keys %{$tempH{$prev_h2}} ) {
                    if ( $boatype_hash{ j_res }{ $bill } eq "" ) {
                        $boatype_hash{ j_res }{ $bill } = 0;
                    }
                    $boatype_hash{ j_res }{ $bill }++;
                }
            }
            elsif( $list[1] eq "Res" ) {
                $res_count++;

                foreach my $bill ( sort keys %{$tempH{$prev_h2}} ) {
                    if ( $boatype_hash{ res }{ $bill } eq "" ) {
                        $boatype_hash{ res }{ $bill } = 0;
                    }
                    $boatype_hash{ res }{ $bill }++;
                }
            }
            else {
              print "$prev_h2\tERROR in recording boatype_hash\n";
            }
        }
        $list = "";
    }
    $prev_h2 = $h2;
```

FIGURE 36D

```
    # SKIP OVER header SECTIONS
#<header>
#<p align="center" bold="on" italic="off"><field><fldinst> PAGE
</fldinst><fldrslt><string charstyname="" bold="on" italic="off">2</string>
#      </fldrslt></field></p>
#<p align="right" bold="on" italic="off"><string bold="on" italic="off">Paul
S. Sarbanes</string></p>
#<p align="left" bold="on" italic="off"><string bold="on" italic="off">SECTION
IV. COSPONSORED MEASURES</string></p>
#<p align="left" bold="on" italic="off"><string bold="on"
italic="off">&tab;&tab;  ORGANIZED BY COMMITTEE REFERRAL</string></p>
#<p align="left" bold="on" italic="off"><string bold="on"
italic="off">&tab;&tab;  SENATE: AGRICULTURE</string></p>
#<p align="left" bold="off" italic="off"></p>
#</header>

    elsif ( $line =~ m/<header/ ) {
        my $i = 0;
        while( $line = <SEN106> ) {
            $line_number++;
            if ( $line =~ m/<p align=.+>(.+)<\/p>/ ) {
                $h4 = $1;
                $i++;
                if ( $i == 3 && ( $section == 2 || $section == 3 || $section
== 4 ) ) {
                    $h4 =~ m/SECTION (.+)\. .+<\/string>/;
                    $h5 = $1;
                    if ( $h5 eq "III" ) {
                        $committee = 3;
                    }
                    elsif ( $h5 eq "IV" ) {
                        $committee = 4;
                    }
                    else {
                        $committee = "";
                    }
                }
                elsif ( $i == 5 && ( $committee == 3 || $committee == 4 ) ) {
                    $h4 =~ m/SECTION (.+)\. .+<\/string>/;
                    $h4 =~ m/&tab;&tab;\s+(.+)<\/string>/;
                    $h6_1 = "COMMITTEE";
                    $h6_2 = "$1";

                    my $ov = $senid . "_" . "$line_number";
#                    $occurrence_value{ $ov } = "";

                    my $len = $#occurrence_committee;
                    $occurrence_committee[ $len + 1 ] = [ ($senid,
$line_number, $committee, $h6_2) ];
                }
            }
```

FIGURE 36E

```
#-----------------------
            my $ov = $senid . "_" . "$line_number";

            $di_value{ $val } = "";

            my $len = $#date_introduced;
            $date_introduced[ $len + 1 ] = [ ($senid, $line_number, $val) ];
#-----------------------

            if ( $section == 1 || $section == 3 ) {
                $tempH{$h2}{SPONSOR} = "$senid";
#-----------------------

                my $ov = $senid . "_" . "-1";

                my $s = $senid;
                $s =~ s/ /_/g;
                $s_value{ $s } = "";

                my $len = $#sponsor;
                $sponsor[ $len + 1 ] = [ ($senid, "-1", $s) ];
#-----------------------
            }

            elsif ( $section == 2 || $section == 4 ) {
                $line = <SEN106>;
                $line_number++;
#                <p bold="off" italic="off">SPONSOR: Daschle</p>
                if ( $line =~ m/<p .+>(.+): (.+)<\/p>/ ) {
                    my $mysponsor = $1;
                    my $value    = $2;
                    $tempH{$h2}{SPONSOR} = $value;
#-----------------------

                    my $ov = $senid . "_" . "$line_number";

                    my $s = $value;
                    $s =~ s/ /_/g;
                    $s_value{ $s } = "";

                    my $len = $#sponsor;
                    $sponsor[ $len + 1 ] = [ ($senid, $line_number, $s) ];
#-----------------------
                }
            }

            elsif ( $section == 5 ) {
#                <p>SUBMITTED FOR: S.    4&tab;CONGRESSIONAL RECORD: S1830</p>
                $line = <SEN106>;
                $line_number++;
                if ( $line =~ m/<p>(.+)<\/p>/ ) {
                    my $submit = $1;
```

FIGURE 36F

```
                my $ov = $senid . "_" . "-1";

                my $s = $senid;
                $s =~ s/ /_/g;
                $s_value{ $s } = "";

                my $len = $#sponsor;
                $sponsor[ $len + 1 ] = [ ($senid, "-1", $s) ];
#-----------------------

                $line = <SEN106>;
                $line_number++;
                if ( $line =~ m/<p>(.+): (.+)<\/p>/ ) {
                    my $submitted_by = $1;
                    my $value = $2;

                    $tempH{$h2}{SUBMITTED_BY} = $value;
#-----------------------

                    my $ov = $senid . "_" . "$line_number";

                    my $sb = $value;
                    $sb =~ s/ /_/g;
                    $sb_value{ $sb } = "";

                    my $len = $#submitted_by;
                    $submitted_by[ $len + 1 ] = [ ($senid, $line_number, $sb)
];
#-----------------------
                }
                elsif ( $line =~ m/<p align=.+>(.+)<\/p>/ ) {
                    $h3 = $1;
                    goto N4;
                }
            }

            elsif ( $section == 6) {
#                <p>SUBMITTED FOR: S.    4&tab;CONGRESSIONAL RECORD: S1830</p>
#                <p>SPONSOR: Murray</p>
#                <p>SUBMITTED BY: Bingaman</p>
                $line = <SEN106>;
                $line_number++;
                if ( $line =~ m/<p>(.+)<\/p>/ ) {
                    my $submit = $1;
                  . my ($part1, $part2) = split(/&tab;/, $submit);
                    my ($part1_1, $part1_2) = split(/: /, $part1);
                    my ($part2_1, $part2_2) = split(/: /, $part2);

                    $part1_2 =~ s/\s*//g;

                    $tempH{$h2}{SUBMITTED_FOR} = $part1_2;
#-----------------------
```

FIGURE 36G

```
#-----------------------
                    $tempH{$h2}{CONGRESSIONAL_RECORD} = $part2_2;
#-----------------------

                    $ov = $senid . "_" . "$line_number";

                    my $cr = $part2_2;
                    $cr =~ s/ /_/g;
                    $cr_value{ $cr } = "";

                    $len = $#congressional_record;
##                    $congressional_record[ $len + 1 ] = [ ($senid,
$line_number, $cr, $h2) ];
                    $congressional_record[ $len + 1 ] = [ ($senid,
$line_number, $cr) ];
#-----------------------
                }
                $line = <SEN106>;
                $line_number++;
                if ( $line =~ m/<p>(.+): (.+)<\/p>/ ) {
                    my $mysponsor = $1;
                    my $value = $2;

                    $tempH{$h2}{SPONSOR} = $value;
#-----------------------

                    my $ov = $senid . "_" . "$line_number";

                    my $s = $value;
                    $s =~ s/ /_/g;
                    $s_value{ $s } = "";

                    my $len = $#sponsor;
                    $sponsor[ $len + 1 ] = [ ($senid, $line_number, $s) ];
#-----------------------
                }
                $line = <SEN106>;
                $line_number++;
                if ( $line =~ m/<p>(.+): (.+)<\/p>/ ) {
                    my $submitted_by = $1;
                    my $value = $2;

                    $tempH{$h2}{SUBMITTED_BY} = $value;
#-----------------------

                    my $ov = $senid . "_" . "$line_number";

                    my $sb = $value;
                    $sb =~ s/ /_/g;
                    $sb_value{ $sb } = "";

                    my $len = $#submitted_by;
```

FIGURE 36H

```
     N4:if ( $h3 eq "COSPONSORS" ) {
#            <p bold="off" italic="off">Edwards; Bayh; Kerry; Bingaman (A-
11/05/1999):</p>
            $line = <SEN106>;
            $line_number++;
            if ( $line =~ m/<p .+>(.+)<\/p>/ ) {
                my $mycosponsors = $1;

                if ( ($mycosponsors ne "") && ($mycosponsors ne "NONE") ) {
                    $tempH{$h2}{COSPONSORS} = $mycosponsors;
#-----------------------

                    my $ov = $senid . "_" . "$line_number";

                    my $cs = $mycosponsors;
                    $cs =~ s/ /_/g;
                    $cs_value{ $cs } = "";

                    my $len = $#cosponsors;
                    $cosponsors[ $len + 1 ] = [ ($senid, $line_number, $cs) ];
#-----------------------
                }
                else {
                    $tempH{$h2}{COSPONSORS} = "NONE";
#-----------------------

                    my $ov = $senid . "_" . "$line_number";

                    my $cs = $mycosponsors;
                    $cs =~ s/ /_/g;
                    $cs_value{ $cs } = "";

                    my $len = $#cosponsors;
                    $cosponsors[ $len + 1 ] = [ ($senid, $line_number, $cs) ];
#-----------------------
                }
            }
            else {
                print LOG "!!! error:$senid:$h2:$h3  COSPONSORS tag !!!!\n";
            }
        }

        elsif ( $h3 eq "OFFICIAL TITLE" ) {
#            <p bold="off" italic="off">.......</p>
            $line = <SEN106>;
            $line_number++;
            if ( $line =~ m/<p .+>(.+)<\/p>/ ) {
                my $title = $1;
                $tempH{$h2}{OFFICIAL_TITLE} = $title;
#-----------------------

            my $ov = $senid . "_" . "$line_number";
```

FIGURE 36I

```
        elsif ( $h3 eq "LATEST STATUS" || $h3 eq "STATUS ACTIONS" ) {
#            <p bold="off"><string bold="on">Oct 25, 1999&tab;Became Public
Law No: 106-80.</string></p>
#            <p><string>May 27, 1999&tab;Proposed amendment S.Amdt. 387
withdrawn in Senate.</string></p>
#            <p><string>May 27, 1999&tab;Proposed by Senator Levin for Senator
Sarbanes.</string></p>
#            <p align="center" italic="off">ABSTRACT</p>
            my $cumulative_content = "";
            my $i = 0;
            my $save_line_number;
            while ( $line = <SEN106> ) {
                $line_number++;
                if ( $i == 0 ) { $save_line_number = $line_number; }
                $i++;
                if ( $line =~ m/<p.*><string.*>(.+)<\/string><\/p>/ ) {
                    my $content = $1;
                    $cumulative_content .= "$content CCCRRR "; # replace \n
with " CCCRRR "
                }
                else {
                    if ( $h3 eq "LATEST STATUS" ) {
                        $tempH{$h2}{LATEST_STATUS} = $cumulative_content;
                    }
                    elsif ( $h3 eq "STATUS ACTIONS" ) {
                        $tempH{$h2}{STATUS_ACTIONS} = $cumulative_content;
                    }
#-----------------------

                    my $ov = $senid . "_" . "$line_number";

                    if ( $h3 eq "LATEST STATUS" ) {
                        my $status = $cumulative_content;
                        $status =~ s/ /_/g;
                        $ls_value{ $status } = "";

                        my $len = $#latest_status;
                        $latest_status[ $len + 1 ] = [ ($senid, $line_number,
$status) ];
                    }
                    elsif ( $h3 eq "STATUS ACTIONS" ) {
                        my $status = $cumulative_content;
                        $status =~ s/ /_/g;
                        $sa_value{ $status } = "";

                        my $len = $#status_actions;
                        $status_actions[ $len + 1 ] = [ ($senid, $line_number,
$status) ];
                    }
#-----------------------
                    goto N2;
                }
```

FIGURE 36J

```
            }
        }

        elsif ( $h3 eq "ABSTRACT" ) {
#             <p italic="off">NONE</p>
            $line = <SEN106>;
            $line_number++;
            if ( $line =~ m/<p.*>(.+)<\/p>/ ) {
                my $abstract = $1;
                $tempH{$h2}{ABSTRACT} = $abstract;
#-----------------------

                my $ov = $senid . "_" . "$line_number";

                my $a = $abstract;
                $a =~ s/ /_/g;
                $a_value{ $a } = "";

                my $len = $#abstract;
                $abstract[ $len + 1 ] = [ ($senid, $line_number, $a) ];
#-----------------------
            }
            else {
                print LOG "!!! error:$senid:$h2:$h3  ABSTRACT tag !!!!\n";
            }
        }

        elsif ( $h3 eq "STATEMENT OF PURPOSE" ) {
#             <p align="center" italic="off">STATEMENT OF PURPOSE</p>

#             <p italic="off">...</p>
#                      OR
#             <p align="center" italic="off">ABSTRACT</p>
#                      OR
#             </section>

            $line = <SEN106>;
            $line_number++;
#-----------------------

            my $ov = $senid . "_" . "$line_number";
#-----------------------
            if ( $line =~ m/<p italic=.+>(.*)<\/p>/ ) {
                my $stmt = $1;
                    if ( $stmt eq "" ) {
                .       $tempH{$h2}{STATEMENT_OF_PURPOSE} = $stmt;
                    }
                    else {
                        $tempH{$h2}{STATEMENT_OF_PURPOSE} = $stmt;
                    }
#-----------------------
                my $sop = $stmt;
                $sop =~ s/ /_/g;
```

FIGURE 36K

```
                my $sop = "";
                $sop_value{ $sop } = "";

                my $len = $#statement_of_purpose;
                $statement_of_purpose[ $len + 1 ] = [ ($senid, $line_number,
$sop) ];
#-----------------------
                goto N2;
            }
            else {
                print LOG "!!! error:$senid:$h2:$h3  STATEMENT OF PURPOSE tag
!!!!\n";
            }

        }

        elsif ( $h3 eq "DIGEST" ) {
#             <p italic="off">NONE</p>
            $line = <SEN106>;
            $line_number++;
            if ( $line =~ m/<p .+>(.+)<\/p>/ ) {
                my $mydigest = $1;
                $tempH{$h2}{DIGEST} = $mydigest;
#-----------------------

                my $ov = $senid . "_" . "$line_number";

                my $d = $mydigest;
                $d =~ s/ /_/g;
                $d_value{ $d } = "";

                my $len = $#digest;
                $digest[ $len + 1 ] = [ ($senid, $line_number, $d) ];
#-----------------------
            }
            else {
                print LOG "!!! error:$senid:$h2:$h3  DIGEST tag !!!!\n";
            }
        }

        else {
            print LOG "!!! error:$senid:$h2:$h3  UNKNOWN tag !!!!\n";
        }
    }
} # END WHILE

THEEND:;
    $prev_h2 = $h2;

} # comment out foreach loop


my $buff = "";
```

FIGURE 36L

```
   PROCS:

     if( $inputstr eq "q" )        { goto MYEND; }
     elsif( $inputstr eq "b" )     { &print_table( "boa" ); }
     elsif( $inputstr eq "a" )     { &print_table( "abstract" ); }
     elsif( $inputstr eq "ha" )    { &print_hash ( "a_value" ); }
     elsif( $inputstr eq "cr" )    { &print_table( "congressional_record" ); }
     elsif( $inputstr eq "hcr" )   { &print_hash ( "cr_value" ); }
     elsif( $inputstr eq "cs" )    { &print_table( "cosponsors" ); }
     elsif( $inputstr eq "hcs" )   { &print_hash ( "cs_value" ); }
     elsif( $inputstr eq "di" )    { &print_table( "date_introduced" ); }
     elsif( $inputstr eq "hdi" )   { &print_hash ( "di_value" ) }
     elsif( $inputstr eq "d" )     { &print_table( "digest" ); }
     elsif( $inputstr eq "hd" )    { &print_hash ( "d_value" ); }
     elsif( $inputstr eq "ls" )    { &print_table( "latest_status" ); }
     elsif( $inputstr eq "hls" )   { &print_hash ( "ls_value" ); }
     elsif( $inputstr eq "sa" )    { &print_table( "status_actions" ); }
     elsif( $inputstr eq "hsa" )   { &print_hash ( "sa_value" ); }
     elsif( $inputstr eq "ot" )    { &print_table( "official_title" ); }
     elsif( $inputstr eq "hot" )   { &print_hash ( "ot_value" ); }
     elsif( $inputstr eq "s" )     { &print_table( "sponsor" ); }
     elsif( $inputstr eq "hs" )    { &print_hash ( "s_value" ); }
     elsif( $inputstr eq "sop" )   { &print_table( "statement_of_purpose" ); }
     elsif( $inputstr eq "hsop" )  { &print_hash ( "sop_value" ); }
     elsif( $inputstr eq "sb" )    { &print_table( "submitted_by" ); }
     elsif( $inputstr eq "hsb" )   { &print_hash ( "sb_value" ); }
     elsif( $inputstr eq "sf" )    { &print_table( "submitted_for" ); }
     elsif( $inputstr eq "hsf" )   { &print_hash ( "sf_value" ); }
     elsif( $inputstr eq "ti" )    { &print_table( "topic_index" ); }
     elsif( $inputstr eq "hti" )   { &print_hash(  "ti_value" ); }

     elsif( $inputstr eq "1" )     { &print_table( "filename_senator" ); }
     elsif( $inputstr eq "2" )     { &print_table( "senator" ); }
     elsif( $inputstr eq "3" )     { &print_table( "filename_period" ); }
     elsif( $inputstr eq "4" )     { &print_table( "prepared_by" ); }
     elsif( $inputstr eq "5" )     { &print_table( "occurrence_section" ); }
     elsif( $inputstr eq "6" )     { &print_table( "occurrence_committee" ); }
     elsif( $inputstr eq "pp" )    { &pretty_print_tables( "106" ); }
     else { print "\t\t**** WRONG OPTION ****\n"; }

     &print_prompt;
}

MYEND:;


close LOG;
exit;


#==========================================================================
====
```

FIGURE 36M

```
    print "\t\"d\".   digest ................. \"hd\".   d_value\n";
    print "\t\"ls\".  latest_status ........... \"hls\".  ls_value\n";
    print "\t\"sa\".  status_actions .......... \"hsa\".  sa_value\n";
    print "\t\"ot\".  official_title .......... \"hot\".  ot_value\n";
    print "\t\"s\".   sponsor ................. \"hs\".   s_value\n";
    print "\t\"sop\". statement_of_purpose .... \"hsop\". sop_value\n";
    print "\t\"sb\".  submitted_by ............ \"hsb\".  sb_value\n";
    print "\t\"sf\".  submitted_for ........... \"hsf\".  sf_value\n";
    print "\t\"ti\".  topic_index ............. \"hti\".  ti_value\n";
    print "\t\n";
    print "\t\"1\". filename_senator     --- \"2\". senator\n";
    print "\t\"3\". filename_period      --- \"4\". prepared_by\n";
    print "\t\"5\". occurrence_section   --- \"6\". occurrence_committee\n";
    print "\t\n";
    print "\t\"pp\". pretty print (Prolog)---\n";
    print " *********************************\n";
}

sub pretty_print_tables {
    my $sid = $_[0];

    my $DIR = "Prolog/sen_$senid";
    mkdir $DIR, 0755;

    open( PP, ">$DIR/boa.P" ) || die "*ERROR: can't open\n" ;
    &pretty_print( "boa" );
    close PP;

    open( PP, ">$DIR/abstract.P" ) || die "*ERROR: can't open\n" ;
    &pretty_print( "abstract" );
    close PP;

    open( PP, ">$DIR/congressional_record.P" ) || die "*ERROR: can't open\n" ;
    &pretty_print( "congressional_record" );
    close PP;

    open( PP, ">$DIR/cosponsors.P" ) || die "*ERROR: can't open\n" ;
    &pretty_print( "cosponsors" );
    close PP;

    open( PP, ">$DIR/date_introduced.P" ) || die "*ERROR: can't open\n" ;
    &pretty_print( "date_introduced" );
    close PP;

    open( PP, ">$DIR/digest.P" ) || die "*ERROR: can't open\n" ;
    &pretty_print( "digest" );
    close PP;

    open( PP, ">$DIR/latest_status.P" ) || die "*ERROR: can't open\n" ;
    &pretty_print( "latest_status" );
    close PP;

    open( PP, ">$DIR/status_actions.P" ) || die "*ERROR: can't open\n" ;
```

FIGURE 36N

```
    open( PP, ">$DIR/statement_of_purpose.P" ) || die "*ERROR: can't open\n" ;
    &pretty_print( "statement_of_purpose" );
    close PP;

    open( PP, ">$DIR/submitted_by.P" ) || die "*ERROR: can't open\n" ;
    &pretty_print( "submitted_by" );
    close PP;

    open( PP, ">$DIR/submitted_for.P" ) || die "*ERROR: can't open\n" ;
    &pretty_print( "submitted_for" );
    close PP;

    open( PP, ">$DIR/topic_index.P" ) || die "*ERROR: can't open\n" ;
    &pretty_print( "topic_index" );
    close PP;

    open( PP, ">$DIR/filename_senator.P" ) || die "*ERROR: can't open\n" ;
    &pretty_print( "filename_senator" );
    close PP;

    open( PP, ">$DIR/filename_period.P" ) || die "*ERROR: can't open\n" ;
    &pretty_print( "filename_period" );
    close PP;

    open( PP, ">$DIR/senator.P" ) || die "*ERROR: can't open\n" ;
    &pretty_print( "senator" );
    close PP;

    open( PP, ">$DIR/prepared_by.P" ) || die "*ERROR: can't open\n" ;
    &pretty_print( "prepared_by" );
    close PP;

    open( PP, ">$DIR/occurrence_section.P" ) || die "*ERROR: can't open\n" ;
    &pretty_print( "occurrence_section" );
    close PP;

    open( PP, ">$DIR/occurrence_committee.P" ) || die "*ERROR: can't open\n" ;
    &pretty_print( "occurrence_committee" );
    close PP;
}

sub pretty_print {
    my $arg_table = $_[0];

# digest('quoted strings', ..., .. , ..).

    my $buff = "";
    my @arr = "";
    my $inputstr = "";

    no strict;
```

FIGURE 36O

```
                else {
                    print PP "\'$aref->[$j]\',";
                }
            }
            $aref->[$n] =~ s/_/ /g;

            if( $arg_table eq "latest_status" || $arg_table eq "status_actions" )
{
                $aref->[$n] =~ s/ CCCRRR /\n/g;
                my @list = split(/\n/, $aref->[$n]);
                my $len = $#list;
                my $newstr = "[";
                foreach my $i ( 0 .. $len ) {
                    my ($ls_date, $ls_mesg) = split(/&tab;/, $list[$i]);
                    $newstr .= "d(\'$ls_date\',";
                    $newstr .= "\'$ls_mesg\')";
                    if( $len > 0 && $i < $len ) {
                        $newstr .= ", ";
                    }
                }
                $newstr .= "]";
                $aref->[$n] = $newstr;
                print PP "$aref->[$n]";
                print PP ")\.\n";
            }

            elsif( $arg_table eq "cosponsors" ) {
#Dodd; Bryan; Leahy; Edwards; Hollings; Breaux (A-02/08/2000):

                if ( $aref->[$n] =~ m/.+:/ ) {
                    chop $aref->[$n];
                }
                my @colist = split(/;/, $aref->[$n]);
                my $newstr = "[";
                foreach my $item (@colist) {
                    my $items = "";
                    $item =~ m/\s*(.+)/ && ($items = $1);
                    my $co_name = "";
                    my $co_amend = "";
#               Mikulski (A-11/08/1999)
                    if ( $items =~ m/(.+) \((.+)\)/ ) {
                        $co_name = $1;
                        $co_amend = $2;
                        $newstr .= "d(\'$co_name\',\'$co_amend\'), ";
                    }
                    else {
                        my $cosponsor_val = $items;
                        $newstr .= "\'$cosponsor_val\', ";
                    }
                }
                chop $newstr;
                chop $newstr;
                $newstr .= "]";
```

FIGURE 36P

```
#============================================================================
====
sub print_bills {
    my $bill;
    my $field;
    foreach $bill ( sort keys %HoH ) {
        print "<bill name=\"$bill\">\n";
        my $flag_NONE = "false";
        foreach $field ( sort keys %{ $HoH{$bill} } ) {
            if( $field eq "DATE_INTRODUCED" ) {
                print "
<date_introduced>$HoH{$bill}{$field}</date_introduced>\n";
            }

#--------------------------
            elsif( $field eq "SPONSOR" ) {
                my $sponsor_val = $HoH{$bill}{$field};
                my $val = "";
                my $print_field;
                $sponsor_val =~ m/^(\d+)/ && ($val = $1);
                if ( $val ne "" ) {  # then it's a number
                    my $key = $senNUMHash{ $val };
                    $print_field = $keypeopleHash{ $key }{ nameURI }; # use of
uninitialized value!!!
                    print "    <sponsor>$print_field</sponsor>\n";
                }
                else {
                    $sponsor_val =~ s/ //g;
                    $sponsor_val = lc $sponsor_val;
                    if ( exists( $keypeopleHash{ $sponsor_val } ) ) {
                        $print_field = $keypeopleHash{ $sponsor_val }{ nameURI
};

                        print "    <sponsor>$print_field</sponsor>\n";
                    }
                    else {
                        print LOG "!!! In print_bills: in SPONSOR section:
keypeopleHash{ $sponsor_val } DOES NOT EXIST!\n";
                        print "    <sponsor>$HoH{$bill}{$field}</sponsor>\n";
                    }
                }
            }
#--------------------------
                .
            elsif( $field eq "COSPONSORS" ) {
                my $cosponsors = $HoH{$bill}{$field};
                if ( $cosponsors =~ m/.+:/ ) {
                    chop $cosponsors;
                } #=== COMMENT if you want COSPONSOR=NONE to disappear
                    print "    <cosponsors>\n";
```

FIGURE 36Q

```
                    if ( $items =~ m/(.+) \((.+)\)/ ) {
                        $co_name = $1;
                        $co_amend = $2;
                        my $cosponsor_val = $co_name;
                        my $val = "";
                        my $print_field;
                        $cosponsor_val =~ s/ //g;
                        $cosponsor_val = lc $cosponsor_val;
                        if ( exists( $keypeopleHash{ $cosponsor_val } ) )
{
                            $print_field = $keypeopleHash{ $cosponsor_val
}{ nameURI };
                            print "            <co_name a-
date=\"$co_amend\">$print_field</co_name>\n";
                        }
                        else {
                            print LOG "!!! In print_bills: in COSPONSORS
section: keypeopleHash{ $cosponsor_val } DOES NOT EXIST!\n";
                            print "            <co_name a-
date=\"$co_amend\">$co_name</co_name>\n";
                        }
                    }
                    else {
                        my $cosponsor_val = $items;
                        my $val = "";
                        my $print_field;
                        $cosponsor_val =~ s/ //g;
                        $cosponsor_val = lc $cosponsor_val;
                        if ( exists( $keypeopleHash{ $cosponsor_val } ) )
{
                            $print_field = $keypeopleHash{ $cosponsor_val
}{ nameURI };
                            print "
<co_name>$print_field</co_name>\n"
                        }
                        else {
                            print LOG "!!! In print_bills: in COSPONSORS
section: keypeopleHash{ $cosponsor_val } DOES NOT EXIST!\n";
                            print "
<co_name>$items</co_name>\n"
                        }
                    }
                    print "        </cosponsor>\n";
                }
                print "    </cosponsors>\n";
#                 )     === UNCOMMENT if you want COSPONSOR=NONE to disappear
            }
            elsif( $field eq "OFFICIAL_TITLE" ) {
                print "
<official_title>$HoH{$bill}{$field}</official_title>\n";
            }

            elsif( $field eq "LATEST_STATUS" ) {
```

FIGURE 36R

```
            elsif( $field eq "ABSTRACT" ) {
                print "    <abstract>$HoH{$bill}{$field}</abstract>\n";
            }
            elsif( $field eq "COMMITTEE" ) {
                my $len = $#{ $HoH{$bill}{COMMITTEE} };
                print "    <committees>\n";
                foreach my $i ( 0 .. $len ) {
                    print "
<committee>$HoH{$bill}{$field}[$i]</committee>\n";
                }
                print "    </committees>\n";
            }
            elsif( $field eq "SUBMITTED_FOR" ) {
                print "
<submitted_for>$HoH{$bill}{$field}</submitted_for>\n";
            }
            elsif( $field eq "CONGRESSIONAL_RECORD" ) {
                print "
<congressional_record>$HoH{$bill}{$field}</congressional_record>\n";
            }

#-------------------------
            elsif( $field eq "SUBMITTED_BY" ) {
                my $submitted_by_val = $HoH{$bill}{$field};
                my $val = "";
                my $print_field;
                    $submitted_by_val =~ s/ //g;
                    $submitted_by_val = lc $submitted_by_val;
                    if ( exists( $keypeopleHash{ $submitted_by_val } ) ) {
                        $print_field = $keypeopleHash{ $submitted_by_val }{
nameURI };
                        print "
<submitted_by>$print_field</submitted_by>\n";
                    }
                    else {
                        print LOG "!!! In print_bills: in SUBMITTEED_BY
section: keypeopleHash{ $submitted_by_val } DOES NOT EXIST!\n";
                        print "
<submitted_by>$HoH{$bill}{$field}</submitted_by>\n";
                    }
            }

            elsif( $field eq "STATEMENT_OF_PURPOSE" ) {
                print "
<statement_of_purpose>$HoH{$bill}{$field}</statement_of_purpose>\n";
            }
            elsif( $field eq "DIGEST" ) {
                print "    <digest>$HoH{$bill}{$field}</digest>\n";
            }
            else {
                print LOG "!!! WRONG TAG:$field !!! in sub: print_bills\n";
            }
        }
```

FIGURE 36S

```
    no strict;

    my $len = $#{$arg_table} + 1;
    print "$arg_table: COUNT=$len\n";
    for my $i ( 0 .. $#{$arg_table} ) {

        if ( !($i % $MORE_SIZE ) && !($i == 0) ) {
            my $percent = ( ($i+1) / $len ) * 100;
            print "--More--";
            printf( "(%1d)", $percent );
            print "\% -- q: to quit\n";
            read(STDIN, $buff, 1);

            for ( my $i = 0; ; read(STDIN, $buff, 1) )  {
                if ( $i == 0 ) { @arr=""; }
                $arr[$i++] = $buff;
                if ( $buff eq "\n" ) {
                    $inputstr = join '', @arr;
                    chop $inputstr;
                    goto NEXT;
                }
                $buff = "";
            }
        }
  NEXT:
        if( $inputstr eq "q" )  { goto END_PRINT_TABLE; }

        my $aref = ${$arg_table}[$i];
        my $n = @$aref - 1;
        printf( "\t%5d:", $i );
        for my $j ( 0 .. $n ) {
             print "\t$aref->[$j]";
        }
        print "\n";
    }
    END_PRINT_TABLE:;
}

sub print_hash {
    my $arg_hash = $_[0];

    no strict;

    my $buff = "";
    my @arr = "";
    my $inputstr = "";

    my $len = scalar keys %{$arg_hash};
    print "$arg_hash Hash: COUNT=$len\n";
    my $i = 0;
    foreach my $key( sort keys %{$arg_hash} ) {

        if ( !($i % $MORE_SIZE ) && !($i == 0) ) {
```

FIGURE 36T

```
            for ( my $i = 0; ; read(STDIN, $buff, 1) )  {
                if ( $i == 0 ) { @arr=""; }
                $arr[$i++] = $buff;
                if ( $buff eq "\n" ) {
                    $inputstr = join '', @arr;
                    chop $inputstr;
                    goto NEXT2;
                }
                $buff = "";
            }
        }
  NEXT2:
        if( $inputstr eq "q" )  { goto END_PRINT_HASH; }

        $i++;
        print "\t*$key*\n";
    }
    END_PRINT_HASH:;
}
#==============================================================================
====
#<section>
#<p fontname="Courier New" fontsize="20"></p>
#<p align="center" fontname="Courier New" fontsize="28" bold="on">UNITED
STATES SENATE</p>
#<p align="center" fontname="Courier New" fontsize="28" bold="on"></p>
#<p align="center" fontname="Courier New" fontsize="28" bold="on"></p>
#<p align="center" fontname="Courier New" fontsize="28" bold="on"></p>
#<p align="center" fontname="Courier New" fontsize="28" bold="on"></p>
#<p align="left" fontname="Courier New" fontsize="48"
bold="on">__________________________</p>
#<p align="left" fontname="Courier New" fontsize="48" bold="on"></p>
#<p align="left" fontname="Courier New" fontsize="48" bold="on">LEGISLATIVE
ACTIVITIES</p>
#<p align="left" fontname="Courier New" fontsize="48"
bold="on">__________________________</p>
#<p align="left" fontname="Courier New" fontsize="48" bold="on"></p>
#<p align="left" fontname="Courier New" fontsize="36" bold="on">THE
HONORABLE</p>
#<p align="left" fontname="Courier New" fontsize="36" bold="on">PAUL S.
SARBANES</p>
#<p align="left" fontname="Courier New" fontsize="36" bold="on">OF
MARYLAND</p>
#<p align="left" fontname="Courier New" fontsize="36" bold="on"></p>
#<p align="left" fontname="Courier New" fontsize="28" bold="on">FOR THE
PERIOD</p>
#<p align="left" fontname="Courier New" fontsize="28" bold="on">JANUARY 06,
1999 TO MARCH 31, 2000</p>
#<p align="left" fontname="Courier New" fontsize="28" bold="on"></p>
#<p align="left" fontname="Courier New" fontsize="48"
bold="on">___________________________</p>
#<p fontname="Courier New" fontsize="20" bold="on"></p>
```

FIGURE 36U

```
#<p fontname="Courier New" fontsize="20" bold="on"></p>
#<p fontname="Courier New" fontsize="20" bold="on"></p>
#<p fontname="Courier New" fontsize="20" bold="on"></p>
#<p fontname="Courier New" fontsize="20" bold="on"></p>
#<p fontname="Courier New" fontsize="20" bold="on"></p>
#<p align="left" fontname="Courier New" fontsize="20" bold="on">Prepared
by:</p>
#<p align="left" fontname="Courier New" fontsize="20" bold="on">Senate
Computer Center</p>
#<p align="left" fontname="Courier New" fontsize="20" bold="on">Office of the
Sergeant at Arms</p>
#<p align="left" fontname="Courier New" fontsize="20" bold="on">and</p>
#<p align="left" fontname="Courier New" fontsize="20" bold="on">Committee on
Rules and Administration</p>
#</section>

sub process_header {
    while( $line = <SEN106> ) {
        $line_number++;
        if ( $line =~ m/<p align=.+>(.+)<\/p>/ ) {
            $h4 = $1;
            if ( $line =~ m/THE HONORABLE/ ) {

                $line = <SEN106>;
                $line_number++;
                $line =~ m/<p align=.+>(.+)<\/p>/;
                $senNAME= $1;
                my $first_line = $line_number;

                $line = <SEN106>;
                $line_number++;
                $line =~ m/OF (.+)<\/p>/;
                $state = $1;

                my @senlist = split(/ /, $senNAME);
                my $firstname = $senlist[0];
                my $lastname = $senlist[$#senlist];
                my $middlepart = "";
                foreach my $i (1 .. ($#senlist-1)) {
                    $middlepart .= "$senlist[$i] ";
                }
                chop $middlepart;

                $len = $#senator;
                $senator[ $len + 1 ] = [ ($senid, $first_line, $firstname,
$middlepart, $lastname, $state) ];


                $line = <SEN106>;
                $line_number++;
                $line = <SEN106>;
                $line_number++;
```

FIGURE 36V

```
                $line = <SEN106>;
                $line_number++;

                $line = <SEN106>;
                $line_number++;

                while( $line = <SEN106> ) {
                    $line_number++;
                    if ( $line =~ m/<p align=/ ) {
                        goto BEGIN_Prepared_by;
                    }
                }
           BEGIN_Prepared_by:;
                my $first_line2 = $line_number;
                $line =~ m/>(.+)<\/p>/;
                my $cumulative_content = "$1 ";

                $line = <SEN106>;
                $line_number++;
                $line =~ m/>(.+)<\/p>/;
                $cumulative_content .= "$1 ";

                $line = <SEN106>;
                $line_number++;
                $line =~ m/>(.+)<\/p>/;
                $cumulative_content .= "$1 ";

                $line = <SEN106>;
                $line_number++;
                $line =~ m/>(.+)<\/p>/;
                $cumulative_content .= "$1 ";

                $line = <SEN106>;
                $line_number++;
                $line =~ m/>(.+)<\/p>/;
                $cumulative_content .= "$1";

                my $ov = $senid . "_" . "$line_number";

                $len = $#prepared_by;
                $prepared_by[ $len + 1 ] = [ ($senid, $first_line2,
$cumulative_content) ];
            }
            $h4 = $1;
        }
        elsif ( $line =~ m/<\/section>/ ) {
            goto PH1;
        }
    }

 PH1:;
    if ( $filename eq "D_1_LARI_S272_106.xml" ) {
```

FIGURE 36W

```
    while( $line = <SEN106> ) {
        $line_number++;
        if ( $line =~ m/<\/section>/ ) {
            goto END_process_header;
        }
    }
    END_process_header:;

}


# <p><string italic="off" hidden="on">SECTION VII. SUBJECT
INDEX</string>ACADEMIC PERFORMANCE&tab; S.7, S.514, S.564</p>
# <p>ACCESS TO HEALTH CARE&tab; S.6, S.1678, S.1690</p>

sub process_index_old {
            $line =~ m/<p>.+<\/string>(.+)&tab; (.+)<\/p>/;
            my $subject = $1;
            my $bill_seq = $2;

#<p align="right">Administrative procedure--Department of Health and Human
Services&tab; S.331, S.1327</p>
#<p>AGED&tab; S.10, S.51, S.331, S.391, S.472, S.718, S.784, S.792,</p>
#<p align="right"> S.1023, S.1074, S.1142, S.1327, S.1499, S.1678, S.1760</p>

        N3:while( $line = <SEN106> ) {
                $line_number++;
                if ( $line =~ m/<p.*>(.+)&tab;(.+\d)(.*)<\/p>/ ) {
                    my $subject = $1;
                    my $bill_seq = $2;
                    my $comma = $3;
                    # IS THERE A CONTINUATION...
                    while ( $comma eq "," ) {
                        my $buf = <SEN106>;
                        $line_number++;
                        $buf=~ m/<p.*>(.+\d)(.*)<\/p>/;
                        my $bill_seq = $1;
                        my $comma = $2;
                        if ( $comma eq "" ) {
                            goto N3;
                        }
                    }
                }
#<p></p>
                elsif ( $line =~ m/<p><\/p>/ ) {
                }
                else {
                    goto THEEND;
                }
            }
}
# <p><string italic="off" hidden="on">SECTION VII. SUBJECT
INDEX</string>ACADEMIC PERFORMANCE&tab; S.7, S.514, S.564</p>
```

FIGURE 36X

```
#<p align="left" fontname="Courier New" fontsize="20" bold="on">Committee on
Rules and Administration</p>
#</section>
#<section>
#<header>
#<p align="center" fontname="Courier New" fontsize="20"
bold="on"><field><fldinst> PAGE </fldinst><fldrslt><string charstyname=""
#            fontname="Courier New" fontsize="20"
bold="on">2</string></fldrslt></field></p>
#<p align="right" fontname="Courier New" fontsize="20" bold="on">
#            <string fontname="Courier New" fontsize="20" bold="on">Lincoln D.
Chafee</string></p>
#<p align="left" fontname="Courier New" fontsize="20" bold="on"><string
fontname="Courier New" fontsize="20" bold="on">
#                  SUBJECT INDEX TO SPONSORED AND COSPONSORED MEASURES AND
AMENDMENTS</string></p>
#<p align="left" fontname="Courier New" fontsize="20" bold="off"></p>
#</header>
#<p>ACCESS TO HEALTH CARE&tab; S.494</p>
#<p>ACCIDENT PREVENTION&tab; S.149, S.936</p>

sub process_index {

            $line =~ m/<p>.+<\/string>(.+)&tab; (.+)<\/p>/;
            my $subject = $1;
            my $bill_seq = $2;
            my @bill_list = split( /,/, $bill_seq );

            my $sub = $subject;
            $sub =~ s/ /_/g;
            $ti_value{ $sub } = "";

            foreach my $item (@bill_list) {
                $item =~ s/\s*//g ;

                my $len = $#topic_index;
                $topic_index[ $len + 1  ] = [ ($senid, $line_number, $sub,
$item) ];
            }

#<p align="right">Administrative procedure--Department of Health and Human
Services&tab; S.331, S.1327</p>
#<p>AGED&tab; S.10, S.51, S.331, S.391, S.472, S.718, S.784, S.792,</p>
#<p align="right"> S.1023, S.1074, S.1142, S.1327, S.1499, S.1678, S.1760</p>

         N3:while( $line = <SEN106> ) {
                $line_number++;
                if ( $line =~ m/<p.*>(.+)&tab;(.+\d)(.*)<\/p>/ ) {
                    my $subject = $1;
                    my $bill_seq = $2;
                    my $comma = $3;
                    my @bill_list = split( /,/, $bill_seq );
```

FIGURE 36Y

```
                # IS THERE A CONTINUATION...
                while ( $comma eq "," ) {
                    my $buf = <SEN106>;
                    $line_number++;
                    $buf=~ m/<p.*>(.+\d)(.*)<\/p>/;
                    my $bill_seq = $1;
                    my $comma = $2;
                    my @bill_list = split( /,/, $bill_seq );
                    foreach my $item (@bill_list) {
                        $item =~ s/\s*//g ;
                        my $len = $#topic_index;
                        $topic_index[ $len + 1  ] = [ ($senid,
$line_number, $subject, $item) ];
                    }
                    if ( $comma eq "" ) {
                        goto N3;
                    }
                }
            }
#<p></p>
            elsif ( $line =~ m/<p><\/p>/ ) {
            }
            else {
                goto THEEND;
            }
        }
}
```

FIGURE 36Z

```
<!ELEMENT SLA_collection =      (senate_file*)>
<!ELEMENT senate_file =    (filename, header_page?, section*, subject_index?)>
<!ELEMENT header_page = (senator?, report_period?, prepared_by?)>
<!ELEMENT senator =             (first_name?, middle_part?, last_name?, state?)>
<!ELEMENT report_date =   (start_date?, end_date?)>
<!ELEMENT section =             (sec_number, sec_name, bar*)>
<!ELEMENT bar =            (bill | resolution | amendment)>
<!ELEMENT resolution =     (joint_resolution, concurrent_resolution, simple_resolution)
<!ELEMENT bill =           (bar_id, date_introduced, sponsor?, cosponsors?, official_title,
                                (latest_status | status_actions), abstract, committee?)>
<!ELEMENT amendment =  (bar_id, date_introduced, submitted_for, congressional_record,
                                sponsor?, submitted_by?, cosponsors,
                                statement_of_purpose, (latest_status | status_actions),
                                abstract)>
```

FIGURE 37

```
#!/usr/local/bin/perl -w
use strict;

# caccf_xml FILE REC_SIZE

my $REC_SIZE = 164;

die "*** Record size must be 164!\n" if ( $ARGV[1] != 164 );

my $rec_count = 0;
my $rec;
my $fsize;



open(IN, "< $ARGV[0]") || die "can't read from: $!";

$fsize = -s IN;

print "*** WARNING: file size ($fsize) is no multiple of record size
($REC_SIZE)\n"
  if ( $fsize % $REC_SIZE != 0 );

open(OUT,"> $ARGV[0].xml") || die "can't write to: $!";
open(LOG,"> $ARGV[0].xml.log") || die "can't write to: $!";

print OUT "<caccf_records>\n";

while (read(IN, $rec, $REC_SIZE) == $REC_SIZE) {
  print OUT "<rec no=\"", ++ $rec_count, "\" ";
  print OUT "ms=\"",substr($rec,0,1),"\" ";
  print OUT "cc=\"",substr($rec,1,2),"\" ";
  print OUT "tc=\"",substr($rec,3,2),"\" ";
  print OUT "rn=\"",substr($rec,5,5),"\" ";
  print LOG "*** ERROR, >", (substr($rec,10,1)), "< instead of blank in #11,
rec $rec_count:\n[[$rec]]\n\n"
    if (substr($rec,10,1) ne " ");
  print OUT "na=\"",substr($rec,11,28),"\" ";
  print OUT "dp=\"",substr($rec,39,4),"\" ";
  print OUT "sn=\"",substr($rec,43,9),"\" ";
  print OUT "mg=\"",substr($rec,52,4),"\" ";
  print OUT "pg=\"",substr($rec,56,2),"\" ";
  print OUT "dd=\"",substr($rec,58,8),"\" ";
  print OUT "hc=\"",substr($rec,66,20),"\" ";
  print OUT "hs=\"",substr($rec,86,2),"\" ";
  print OUT "oc=\"",substr($rec,88,5),"\" ";
  print OUT "db=\"",substr($rec,93,8),"\" ";
  print OUT "rc=\"",substr($rec,101,1),"\" ";
  print OUT "ai=\"",substr($rec,102,1),"\" ";
  print OUT "ra=\"",substr($rec,103,1),"\" ";
  print OUT "re=\"",substr($rec,104,2),"\" ";
  print OUT "le=\"",substr($rec,106,2),"\" ";
  print OUT "ma=\"",substr($rec,108,1),"\" ";
  print OUT "se=\"",substr($rec,109,1),"\" ";
```

FIGURE 38

```
#!/usr/local/bin/perl -w
use strict;

# caccf2oracle FILE REC_SIZE

my $REC_SIZE = 164;

#die "*** Record size must be 164!\n" if ( $ARGV[1] != 164 );

my $rec_count = 0;
my $rec;
my $fsize;


my %month = ("01" => "JAN",
             "02" => "FEB",
             "03" => "MAR",
             "04" => "APR",
             "05" => "MAY",
             "06" => "JUN",
             "07" => "JUL",
             "08" => "AUG",
             "09" => "SEP",
             "10" => "OCT",
             "11" => "NOV",
             "12" => "DEC" );

sub mm_dd_yy {
  my ($a_date) = @_ ;
  if ($a_date =~ m|([0-9]{2})/([0-9]{2})/([0-9]{2})| ) {
    return "TO_DATE('$2-$month{$1}-19$3')";
  } else {
    print "*** ERROR   Rec #$rec_count, mm_dd_yy, not a date >>>$a_date<<<\n";
    return "NULL";
  }
};

sub yymmdd {
  my ($a_date) = @_ ;
  if ($a_date =~ m|([0-9]{2})([0-9]{2})([0-9]{2})| ) {
    "TO_DATE('$3-$month{$2}-19$1')";
  } else {
    print "*** WARNING Rec #$rec_count, yymmdd, not a date >>>$a_date<<<\n";
    return "NULL";
  };
};

sub escapeQuote {
  my ($a_string) = @_ ;
  if ($a_string =~ s/\'/\'\'/g) {
    print "*** NOTE    Rec #$rec_count, quote escaped >>>$a_string<<<\n";
  }
  return $a_string;
}
```

FIGURE 39A

39A
39B

```
open(OUT,"> $ARGV[0].sql") || die "can't write to: $!";
open(LOG,"> $ARGV[0].sql.log") || die "can't write to: $!";


while (read(IN, $rec, $REC_SIZE) == $REC_SIZE) {
  print OUT "insert into CACCF values (";
  print OUT "'",  ++ $rec_count  ,"',"; # internal record number
  print OUT "'",substr($rec,0,1),"',"; # ms
  print OUT "'",substr($rec,1,2),"',"; # cc
  print OUT "'",substr($rec,3,2),"',"; # tc
  print OUT "'",substr($rec,5,5),"',"; # rn
  print OUT "'",escapeQuote(substr($rec,11,28)),"',"; # na
  print OUT "'",substr($rec,39,4),"',";  # dp
  print OUT "'",substr($rec,43,9),"',"; # sn
  print OUT "'",substr($rec,52,4),"',"; # mg
  print OUT "'",substr($rec,56,2),"',"; # pg
  print OUT mm_dd_yy(substr($rec,58,8)),","; # dd
  print OUT "'",escapeQuote(substr($rec,66,20)),"',"; # hc
  print OUT "'",substr($rec,86,2),"',"; # hs
  print OUT "'",substr($rec,88,5),"',"; # oc
  print OUT mm_dd_yy(substr($rec,93,8)),","; # db
  print OUT "'",substr($rec,101,1),"',"; # rc
  print OUT "'",substr($rec,102,1),"',"; # ai
  print OUT "'",substr($rec,103,1),"',"; # ra
  print OUT "'",substr($rec,104,2),"',"; # re
  print OUT "'",substr($rec,106,2),"',"; # le
  print OUT "'",substr($rec,108,1),"',"; # ma
  print OUT "'",substr($rec,109,1),"',"; # se
  print OUT "'",substr($rec,110,1),"',"; # ci
  print OUT "'",substr($rec,111,1),"',"; # pp
  print OUT yymmdd(substr($rec,112,6)),","; # dt
  print OUT "'",substr($rec,118,1),"',"; # lr
  print OUT "'",substr($rec,119,3),"',"; # br
  print OUT "'",substr($rec,122,2),"',"; # ag
  print OUT "'",substr($rec,124,1),"',"; # sc
  print OUT "'",escapeQuote(substr($rec,125,29)),"',"; # co
  print OUT "'",escapeQuote(substr($rec,154,2)),"',"; # ty
  print OUT "'",escapeQuote(substr($rec,156,2)),"',"; # pc
  print OUT "'",substr($rec,158,2),"',"; # mc
  print OUT "'",substr($rec,160,2),"',"; # pr
  print OUT "'",substr($rec,162,2),"'"; # fl
  print OUT ");\n";
};



print LOG "[$ARGV[0]: read ",$rec_count*$REC_SIZE, " bytes = $rec_count
records x $REC_SIZE]\n";

print LOG "*** WARNING: file size = $fsize\n"
  if ( $rec_count*$REC_SIZE != $fsize );
```

FIGURE 39B

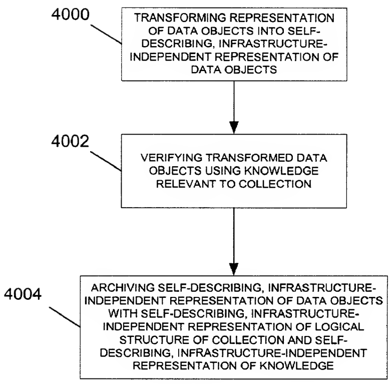
4000
TRANSFORMING REPRESENTATION OF DATA OBJECTS INTO SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT REPRESENTATION OF DATA OBJECTS
4002
VERIFYING TRANSFORMED DATA OBJECTS USING KNOWLEDGE RELEVANT TO COLLECTION
4004
ARCHIVING SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT REPRESENTATION OF DATA OBJECTS WITH SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT REPRESENTATION OF LOGICAL STRUCTURE OF COLLECTION AND SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT REPRESENTATION OF KNOWLEDGE

FIGURE 40

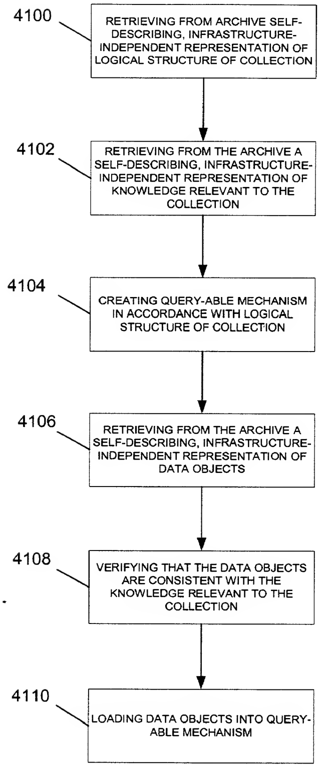
4100
RETRIEVING FROM ARCHIVE SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT REPRESENTATION OF LOGICAL STRUCTURE OF COLLECTION
4102
RETRIEVING FROM THE ARCHIVE A SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT REPRESENTATION OF KNOWLEDGE RELEVANT TO THE COLLECTION
4104
CREATING QUERY-ABLE MECHANISM IN ACCORDANCE WITH LOGICAL STRUCTURE OF COLLECTION
4106
RETRIEVING FROM THE ARCHIVE A SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT REPRESENTATION OF DATA OBJECTS
4108
VERIFYING THAT THE DATA OBJECTS ARE CONSISTENT WITH THE KNOWLEDGE RELEVANT TO THE COLLECTION
4110
LOADING DATA OBJECTS INTO QUERY-ABLE MECHANISM

FIGURE 41A

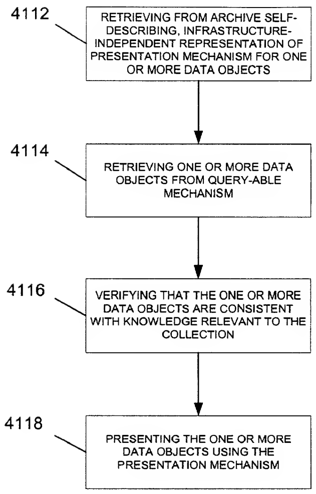
4112
RETRIEVING FROM ARCHIVE SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT REPRESENTATION OF PRESENTATION MECHANISM FOR ONE OR MORE DATA OBJECTS
4114
RETRIEVING ONE OR MORE DATA OBJECTS FROM QUERY-ABLE MECHANISM
4116
VERIFYING THAT THE ONE OR MORE DATA OBJECTS ARE CONSISTENT WITH KNOWLEDGE RELEVANT TO THE COLLECTION
4118
PRESENTING THE ONE OR MORE DATA OBJECTS USING THE PRESENTATION MECHANISM

FIGURE 41B

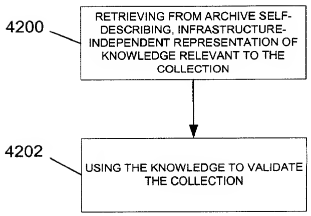
4200
RETRIEVING FROM ARCHIVE SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT REPRESENTATION OF KNOWLEDGE RELEVANT TO THE COLLECTION
4202
USING THE KNOWLEDGE TO VALIDATE THE COLLECTION

FIGURE 42

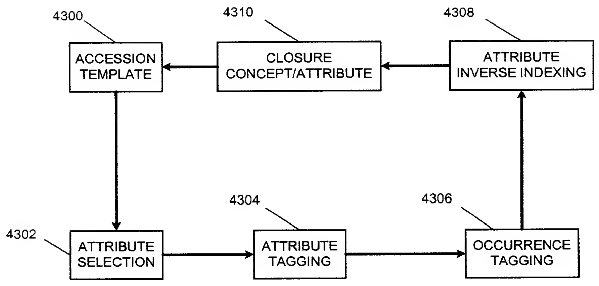
4300
ACCESSION TEMPLATE
4310
CLOSURE CONCEPT/ATTRIBUTE
4308
ATTRIBUTE INVERSE INDEXING
4302
ATTRIBUTE SELECTION
4304
ATTRIBUTE TAGGING
4306
OCCURRENCE TAGGING

FIGURE 43

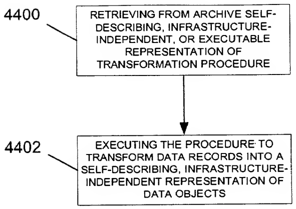
4400
RETRIEVING FROM ARCHIVE SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT, OR EXECUTABLE REPRESENTATION OF TRANSFORMATION PROCEDURE
4402
EXECUTING THE PROCEDURE TO TRANSFORM DATA RECORDS INTO A SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT REPRESENTATION OF DATA OBJECTS

FIGURE 44A

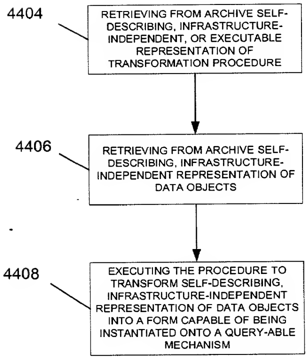
4404
RETRIEVING FROM ARCHIVE SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT, OR EXECUTABLE REPRESENTATION OF TRANSFORMATION PROCEDURE
4406
RETRIEVING FROM ARCHIVE SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT REPRESENTATION OF DATA OBJECTS
4408
EXECUTING THE PROCEDURE TO TRANSFORM SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT REPRESENTATION OF DATA OBJECTS INTO A FORM CAPABLE OF BEING INSTANTIATED ONTO A QUERY-ABLE MECHANISM

FIGURE 44B

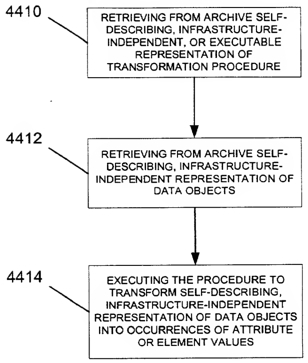
4410
RETRIEVING FROM ARCHIVE SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT, OR EXECUTABLE REPRESENTATION OF TRANSFORMATION PROCEDURE
4412
RETRIEVING FROM ARCHIVE SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT REPRESENTATION OF DATA OBJECTS
4414
EXECUTING THE PROCEDURE TO TRANSFORM SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT REPRESENTATION OF DATA OBJECTS INTO OCCURRENCES OF ATTRIBUTE OR ELEMENT VALUES

FIGURE 44C

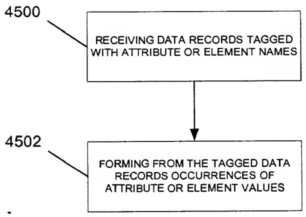
4500
RECEIVING DATA RECORDS TAGGED WITH ATTRIBUTE OR ELEMENT NAMES
4502
FORMING FROM THE TAGGED DATA RECORDS OCCURRENCES OF ATTRIBUTE OR ELEMENT VALUES

FIGURE 45

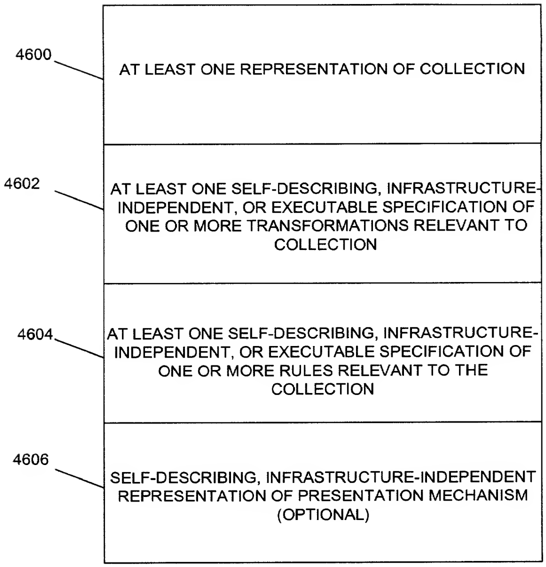
4600
AT LEAST ONE REPRESENTATION OF COLLECTION
4602
AT LEAST ONE SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT, OR EXECUTABLE SPECIFICATION OF ONE OR MORE TRANSFORMATIONS RELEVANT TO COLLECTION
4604
AT LEAST ONE SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT, OR EXECUTABLE SPECIFICATION OF ONE OR MORE RULES RELEVANT TO THE COLLECTION
4606
SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT REPRESENTATION OF PRESENTATION MECHANISM (OPTIONAL)

FIGURE 46

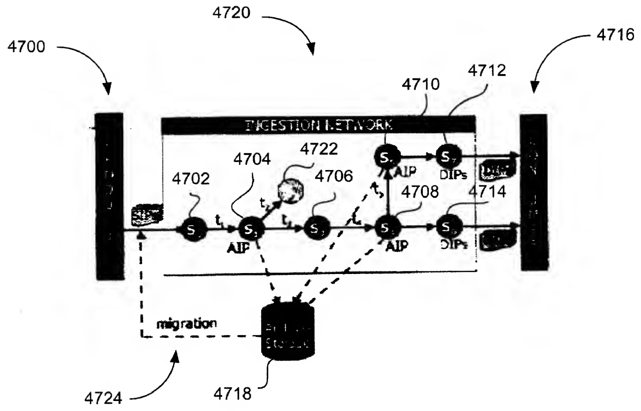
4720
4700
4716
4712
4710
INGESTION NETWORK
4722
4704
4706
4702
4708
4714
AIP
DIPs
SIP
migration
4724
4718

FIGURE 47

```
      %%% Rules for (!ELEMENT X   (Y Z) )
      false ← P : X, not (P 1) : Y                      % 1st child is not Y
4800  false ← P : X, not (P 2) : Z.                     % 2nd child is not Z
      false ← P : X, not P[_→_]                         % there are no children
      false ← P : X[N→_], not N=1, not N=2.             % there are other children
      %%% Rules for (!ELEMENT X   (Y | Z) )
      false ← P : X[1→A], not A : Y, not A : Z          % 1st child other than Y or Z
4802  false ← P : X, not P[_→_].                        % there are no children
      false ← P : X[N→_], not N=1.                      % a non-1st child
4804  %%% Rule for (!ELEMENT X   (Y)* )
      false ← P : X[_→C], not C : Y.                    % a non-Y child
```

FIGURE 48

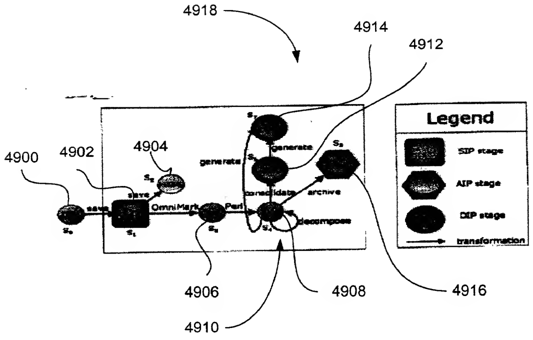
4918
4914
4912
4900
4902
4904
4906
4908
4910
4916
S_0
save
S_1
save
S_2
OmniMark
S_3
Perl
S_4
decompose
consolidate
archive
S_5
generate
S_6
generate
S_7
Legend
SIP stage
AIP stage
DIP stage
transformation

FIGURE 49

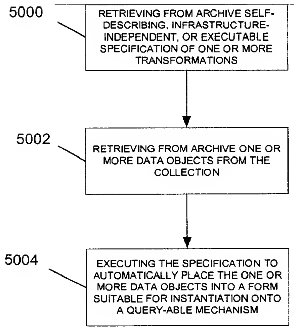
5000
RETRIEVING FROM ARCHIVE SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT, OR EXECUTABLE SPECIFICATION OF ONE OR MORE TRANSFORMATIONS
5002
RETRIEVING FROM ARCHIVE ONE OR MORE DATA OBJECTS FROM THE COLLECTION
5004
EXECUTING THE SPECIFICATION TO AUTOMATICALLY PLACE THE ONE OR MORE DATA OBJECTS INTO A FORM SUITABLE FOR INSTANTIATION ONTO A QUERY-ABLE MECHANISM

FIGURE 50

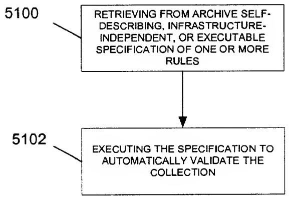
5100
RETRIEVING FROM ARCHIVE SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT, OR EXECUTABLE SPECIFICATION OF ONE OR MORE RULES
5102
EXECUTING THE SPECIFICATION TO AUTOMATICALLY VALIDATE THE COLLECTION

FIGURE 51A

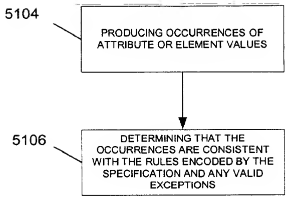
5104
PRODUCING OCCURRENCES OF ATTRIBUTE OR ELEMENT VALUES
5106
DETERMINING THAT THE OCCURRENCES ARE CONSISTENT WITH THE RULES ENCODED BY THE SPECIFICATION AND ANY VALID EXCEPTIONS

FIGURE 51B

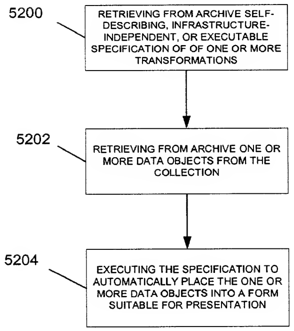
5200
RETRIEVING FROM ARCHIVE SELF-DESCRIBING, INFRASTRUCTURE-INDEPENDENT, OR EXECUTABLE SPECIFICATION OF OF ONE OR MORE TRANSFORMATIONS
5202
RETRIEVING FROM ARCHIVE ONE OR MORE DATA OBJECTS FROM THE COLLECTION
5204
EXECUTING THE SPECIFICATION TO AUTOMATICALLY PLACE THE ONE OR MORE DATA OBJECTS INTO A FORM SUITABLE FOR PRESENTATION

FIGURE 52

```
#-----------------------------------------------------------
# An excerpt of an example of a Topic Map for the SLA (Senate)
# Collection.
#
# 4 Topics are shown:  t1, t2, t3, and t4 of type "SubjectEntry"
# --> These are actually Subject Index Entries found in the
#                                raw data
#
# For each topic, there is an occurence list of locator elements
# corresponding to the bills that discuss that topic.
#
# In addition, topics are related to each other through associations.
# Here we created two types of associations:
#   <assoc types="CoDiscussedInExactlyOneBill">
#   <assoc types="CoDiscussedInTwoOrMoreBills">
#
# showing the "degree of connectedness" between two topics.
# These would be value-added relationships, as they are implicit
# in the raw data, and discovered by our topic map building
# routines.
#
# Bertram Ludaescher & Richard Marciano -- March 20, 2001
#-----------------------------------------------------------

<!DOCTYPE topicmap [
<!ELEMENT topicmap (topic | assoc )* >
<!ELEMENT topic (topname | occurs)* >
<!ATTLIST topic
id      ID    #REQUIRED
types   CDATA #IMPLIED
>
<!ELEMENT topname (basename, dispname, sortname)>
<!ELEMENT basename (#PCDATA) >
<!ELEMENT dispname (#PCDATA) >
<!ELEMENT sortname (#PCDATA) >
<!ELEMENT occurs (locator*) >
<!ELEMENT locator EMPTY >
<!ATTLIST locator
role  CDATA   #REQUIRED
href  CDATA   #REQUIRED
>
<!ELEMENT assoc (assocrl*) >
<!ATTLIST assoc
types  CDATA #IMPLIED
>
<!ELEMENT assocrl EMPTY >
<!ATTLIST assocrl
role   CDATA   #REQUIRED
href   CDATA   #REQUIRED
>
]>
```

54A
54B
54C

FIGURE 54A

```
<topicmap>
<topic id="t1" types="SubjectEntry">
<topname>
<basename>Apartment houses</basename>
<dispname>Apt. Houses</dispname>
<sortname>APARTMENTHOUSES</sortname>
</topname>
<occurs>
<locator role="DiscussedIn" href="#S.463" />
</occurs>
</topic>

<topic id="t2" types="SubjectEntry">
<topname>
<basename>Children</basename>
<dispname>Child.</dispname>
<sortname>CHILDREN</sortname>
</topname>
<occurs>
<locator role="DiscussedIn" href="#S.300" />
<locator role="DiscussedIn" href="#S.463" />
<locator role="DiscussedIn" href="#S.1638" />
<locator role="DiscussedIn" href="#S.1673" />
<locator role="DiscussedIn" href="#S.1709" />
<locator role="DiscussedIn" href="#S.Res.125" />
<locator role="DiscussedIn" href="#S.Res.258" />
</occurs>
</topic>

<topic id="t3" types="SubjectEntry">
<topname>
<basename>Welfare</basename>
<dispname>Welf.</dispname>
<sortname>WELFARE</sortname>
</topname>
<occurs>
<locator role="DiscussedIn" href="#S.463" />
<locator role="DiscussedIn" href="#S.1277" />
<locator role="DiscussedIn" href="#S.1709" />
<locator role="DiscussedIn" href="#S.Con.Res.28" />
<locator role="DiscussedIn" href="#S.Res.125" />
<locator role="DiscussedIn" href="#S.Res.260" />
</occurs>
</topic>
```

FIGURE 54B

```
<topic id="t4" types="SubjectEntry">
<topname>
<basename>Youth employment</basename>
<dispname>Youth empl.</dispname>
<sortname>YOUTEMPLOYMENT</sortname>
</topname>
<occurs>
<locator role="DiscussedIn" href="#S.463" />
</occurs>
</topic>

<assoc types="CoDiscussedInExactlyOneBill">
<assocrl role="DiscussedInSameBill" href="t1" />
<assocrl role="DiscussedInSameBill" href="t2" />
<assocrl role="DiscussedInSameBill" href="t3" />
<assocrl role="DiscussedInSameBill" href="t4" />
</assoc>

<assoc types="CoDiscussedInTwoOrMoreBills">
<assocrl role="DiscussedInSameBill" href="t2" />
<assocrl role="DiscussedInSameBill" href="t3" />
</assoc>

</topicmap>
```

FIGURE 54C